

Application No.: 10/016599

Docket No.: SMQ-066/P5901

AMENDMENTS TO THE SPECIFICATION

Please make the following changes to the title:

OBJECT CLASS FOR FACILITATING CONVERSION BETWEEN JAVA AND XML

Please replace paragraph 1 with the following:

The present invention relates generally to computer programming and more particularly to an object class for facilitating conversion between ~~Java~~JAVA objects and XML.

Please replace paragraph 2 with the following:

~~Java~~The JAVA programming language is a programming language that is object oriented and provides platform independence. (~~Java~~JAVA is a trademark of Sun Microsystems, Inc. in the United States and in other countries). As an object-oriented language, ~~Java~~the JAVA programming language supports the notion of an "object class." An object class describes a group of objects with similar properties, behavior, common relationships to other objects and, semantics. An "object" is an instance of an object class. Each object may encapsulate both methods and data.

Please replace paragraph 4 with the following:

XML and ~~Java~~the JAVA programming language are both popular for use in network applications. For example, web pages and other kinds of content that are available via network applications may entail the use of both ~~Java~~the JAVA programming language and XML. As such, it often may be desirable to convert between ~~Java~~the JAVA programming language and XML. Unfortunately, developers currently are required to generate custom applications to perform this conversion. This approach is time-consuming, prone to error, and laborious.

Please replace paragraph 5 with the following:

Application No.: 10/016599

Docket No.: SMQ-066/P5901

The present invention addresses the above-described limitations of conventional systems. In particular, the present invention provides a conveniently used mechanism for converting from Java/JAVA objects to XML and then back from XML to Java/JAVA objects. The mechanism is generalizable enough to support the conversion of many different varieties of Java/JAVA objects into XML and many different varieties of XML structures into Java/JAVA objects.

Please replace paragraph 6 with the following:

In an illustrative embodiment of the present invention, a base class is defined that includes conversion methods. A first conversion method converts Java/JAVA objects into XML, and a second conversion method converts XML into Java/JAVA objects. A selected object class may then be declared that is a subclass of the base class containing the conversion methods and that extends the base class. Thus, the conversion method merely needs to be invoked in order to perform the conversion of an object of the selected object class.

Please replace paragraph 7 with the following:

In accordance with one aspect of the present invention, a method is practiced in an electronic device. In accordance with this method, a base object class is provided that includes at least one method for converting between Java/JAVA objects and XML data objects. A selected object class is provided that is a subclass of the base object class. For a given instance of the selected object class, the methods invoked perform the conversion of the given instance.

Please replace paragraph 8 with the following:

In accordance with an additional aspect of the present invention, a method is practiced in an electronic device where a base object class is provided. The base object class includes at least one method for converting Java/JAVA objects and XML data objects. The first object class is

Application No.: 10/016599

Docket No.: SMQ-066/P5901

defined as a sub-class of the base object class as is a second object class. The method for converting may be invoked for an instance of the first object class and/or an instance of the second object class. The method may convert a JavaJAVA object in XML or conversely an XML document into JavaJAVA objects.

Please replace paragraph 9 with the following:

An illustrative embodiment of the present invention will be described below relative to the following drawings FIG. 1-depicts the flow of conversion between JavaJAVA objects and XML in the illustrative embodiment.

Please replace paragraph 14 with the following:

The illustrative embodiment of the present invention provides a base object class designated as "XmlBase." The XmlBase object class contains methods for converting from JavaJAVA objects to XML and then back from XML to JavaJAVA objects. In order for an object to utilize the capabilities of the XmlBase object class, the object is defined to be of an object class that is a subclass of the XmlBase object class. The subclass extends the XmlBase object class. The methods for converting between JavaJAVA objects and XML may then be executed on the object.

Please replace paragraph 15 with the following:

FIG. 1 is a block diagram that illustrates the conversion paths that are available between JavaJAVA objects and XML in the illustrative embodiment of the present invention. A JavaJAVA object 10 may be converted into XML 14 by invoking the to XmlDoc() method 12. This method, as will be described in more detail below, takes a JavaJAVA object and converts the JavaJAVA object into structured data in an XML document. The XML document 14 may then be converted back into a JavaJAVA object 10 by invoking the from XmlDoc() method 16.

Application No.: 10/016599

Docket No.: SMQ-066/P5901

Please replace paragraph 16 with the following:

As was mentioned above, the JavaJAVA object 10 is of an object class that is a subclass of the XmlBase object class. As such, the JavaJAVA object inherits the methods of the XmlBase object class. These methods include the to XmlDoc() method 12 and the from XmlDoc() method 16. The object class of the object may be defined to extend the XmlBase object class. For example, suppose that an object is of object class A. In such a case, the object class definition for object class A might contain the statement "public class A extends XmlBase."

Please replace paragraph 17 with the following:

FIG. 2 shows a block diagram of an electronic device 20 that may be suitable for practicing the illustrative embodiment of the present invention. Those skilled in the art will appreciate that the electronic device 20 may take many forms including but not limited to a workstation, a personal computer, a network computer, an Internet appliance, a personal digital assistant ("PDA"), a mobile phone, an intelligent pager or another type of electronic device that is capable of understanding Javathe JAVA programming language and XML. Those skilled in the art will also appreciate that the configuration shown in FIG. 2 is intended to be merely illustrative and not limiting of the present invention. The present invention may be practiced with different configurations that do not include all the peripheral devices depicted in FIG. 2.

Please replace paragraph 19 with the following:

The electronic device 20 includes storage 34 that may include both primary memory and secondary memory. The storage 34 may include computer-readable media and may include removable media, such as a magnetic disk or an optical disk. The storage 34 holds a JavaJAVA compiler 36 for compiling JavaJAVA code into byte codes that may be executed by a JavaJAVA virtual machine (VM) 38. The storage 34 may include a multitude of JavaJAVA objects 40, including a definition for the XmlBase object class 42. The storage additionally includes an XML interpreter 44 and one or more XML documents 46. The storage may include an operating system 48, such as the Solaris operating system available from Sun Microsystems, Inc. Lastly, the storage 34 may include a helper section of code 50 that helps to parse XML and to perform

Application No.: 10/016599

Docket No.: SMQ-066/P5901

other helpful functions.

Please replace paragraph 21 with the following:

JavaJAVA interfaces and classes may be grouped into packages. Sun Microsystems, Inc. has developed and made commercially available a number of different packages. These packages include but are not limited to the java.lang package that contains basic JavaJAVA classes, the java.io package that provides classes to manage input and output streams and the java.util package that contains a number of utility classes. The illustrative embodiment will be described relative to an implementation wherein the XmlBase object class imports these packages.

Please replace paragraph 22 with the following:

The illustrative embodiment also utilizes "reflection" technology from Sun Microsystems, Inc. The reflection technology facilitates the interrogation of JavaJAVA objects to obtain information regarding the objects, such as methods, fields and the like that are contained in the object.

Please replace paragraph 23 with the following:

FIG. 3 is a flow chart illustrating at a high level the steps that are performed by the to XmlDoc() method in the illustrative embodiment. The method initially creates a StringBuffer object class (step 60 in FIG. 3). The StringBuffer object class is defined as part of the java.lang package and implements a mutable sequence of characters. More informally, it provides a buffer for holding strings. A StringBuffer may be created by calling a constructor method that is defined as part of the StringBuffer object class. By definition, the StringBuffer provides an append method that allows characters to be appended to the StringBuffer. The StringBuffer is used to hold characters from the JavaJAVA object that is being converted and to hold tags and other information for creating a properly formatted XML document.

Application No.: 10/016599

Docket No.: SMQ-066/P5901

Please replace paragraph 24 with the following:

The to XmlDoc() method then fills the StringBuffer with the characters extracted from the JavaJAVA object (step 62 in FIG. 3). In particular, the to XmlDoc() method parses the contents of the JavaJAVA object to identify constituent components, such as methods, object classes, fields and the like. XML tags are inserted into the buffer at the appropriate locations so that proper structure is created. Tags to designate the beginning of the XML document and the end of the XML document are also inserted in sequence.

Please replace paragraph 25 with the following:

As was mentioned above, the to XmlDoc() method is applied on a per object basis. The processing initially obtains the object class of the JavaJAVA object. An appropriate tag is provided to designate the object class in the XML document by adding the tag to the StringBuffer. Methods, fields, and other components in the object are then systematically and iteratively located by processing the object on a line per line basis. The reflection API contains methods for obtaining the name, type, arguments, and values of the components. These are used to extract the information that is added to the StringBuffer. Non-string values are converted into strings before being added to the StringBuffer. The appropriate tags are inserted where needed for each component. The tag is the name of the component.

Please replace paragraph 27 with the following:

An XML document is created by invoking the createXmlDocument() method (step 66 in FIG. 3). The java.lang package includes an XmlDocument object class. This object class includes the createXmlDocument() method that constructs an XML document from data in a specified input source. In the illustrative embodiment, the input source is specified as the StringReader that was created in step 64. Hence, in step 68 in FIG. 3, the XML document is filled using the StringReader. The resulting XML document is returned as the converted document holding the XML version of the JavaJAVA object.

Application No.: 10/016599

Docket No.: SMQ-066/P5901

Please replace paragraph 32 with the following:

FIG. 4 provides a high-level flow chart of steps performed by the from XmlDoc() method. As was mentioned above, this method converts an XML document created by the to XmlDoc() method into a JavaJAVA object. Initially, child nodes of the document root node in the XML document are obtained (step 80 in FIG. 4). It is likely that one of these child nodes holds information regarding the object class associated with the XML document. In the illustrative embodiment, it is presumed that the XML document was created by converting a JavaJAVA object. Thus, in step 82, the class of the object is located. The nodes are then processed (step 84 in FIG. 4). In particular, the nodes on the node list that form the XML document are processed. The processing converts the nodes into an appropriate JavaJAVA object. The resulting JavaJAVA object that is created by processing all of the nodes is then returned (step 86 in FIG. 4).

Please replace paragraph 34 with the following:

From the gathered information, the node is processed. In particular, the fields that are located within the node are identified and converted into the appropriate JavaJAVA statements. Other methods and objects are also processed to determine their type and develop the appropriate JavaJAVA syntax. All of this information is encapsulated into an object that is available and eventually encapsulated into the JavaJAVA object that is returned by from XmlDoc() (step 98 in FIG. 5).

Please replace paragraph 35 with the following:

The from XmlDoc() method exploits the XML document complying with DOM. In particular, the method walks the node list and processes nodes individually. The helper code 50 may be used to parse XML tags and other content. The method identifies the type of information contained in the node and creates the appropriate JavaJAVA statement. The objects associated with a node may be of many different data types and these data types are identified.

Please make the following changes to the Abstract:

Application No.: 10/016599

Docket No.: SMQ-066/P5901

A base object class is defined for conversion between ~~Java~~JAVA objects and XML. The base object class may include a first method for converting from ~~Java~~JAVA objects to XML. The base object class may also include a second method for converting from XML to ~~Java~~JAVA objects. Object classes may then be defined that extend the base object class and that facilitate conversion between ~~Java~~JAVA objects and XML.